AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A computer-implemented method of dynamically generating web pages, said method comprising:

analyzing a page that includes static markup text and a set of code instructions executable on a server;

extracting the static markup text from the page and storing the static markup text in a resource file;

generating a servlet class for the page based on the set of code instructions, wherein the servlet class comprises a static initializer for initializing a static class variable for the servlet class, and wherein the servlet class does not include the static markup text;

in response to a first use of the servlet class by any instance of the servlet class:

invoking the static class initializer of the servlet class, wherein the invoking of the static class initializer of the servlet class causes the static markup text to be read from the resource file, and the static class variable to be initialized with the static markup text;

loading a copy of the static class variable initialized with the static markup text into shared memory;

in response to each request of a plurality of requests for the page from a plurality of clients, performing the steps of:

instantiating a distinct instance of the servlet class on the server, wherein instantiating each instance of the servlet class does not create another copy of the static markup text;

executing said distinct instance of the servlet class, wherein execution of each instance of the server class generates a compiled page ~~based on the copy of~~ by combining the static markup text that resides in the shared memory~~, and~~ with results produced by executing the set of code instructions; and

sending the compiled page to a client that requested the page;

wherein the method is performed by one or more computing devices.

2. - 4 (Canceled)

5. (Currently Amended) A method of initiating a first instance of an application that shares a set of static markup text with other instances of the application, wherein the first instance of the application is generated by compiling code from a page that contains both the code and the set of static markup text in response to a request from one or more users, said method comprising:

executing instructions to instantiate the first instance of the application, wherein said instructions are stored on a non-transitory computer-readable storage medium, said instructions that, when executed, cause one or more processors to perform the steps of:

analyzing the page to extract the set of static markup text and storing the set of static markup text in a resource file;

generating a servlet class for the page based on the code from the page, wherein the servlet class comprises a static initializer for initializing a set of static class variables for the servlet class, and wherein the servlet class does not include the set of static markup text;

in response to a first use of the servlet class by any instance of the servlet class:

invoking the static class initializer of the servlet class, wherein the invoking of the static class initializer of the servlet class causes the set of static markup text to be read from the resource file, and the set of static class variables to be initialized with the set of static markup text;

loading a copy of the set of static class variables initialized with the set of static markup text into shared, read-only memory;

in response to each request of a plurality of requests for the page from the one or more users, performing the steps of:

instantiating a distinct instance of the servlet class, wherein instantiating each instance of the servlet class does not create another copy of the set of static markup text;

executing said distinct instance of the servlet class, wherein execution of each instance of the server class generates a compiled page ~~based on the copy of~~ by combining the set of static markup text that resides in the shared, read-only memory~~, and~~ with results produced by executing the set of code instructions;

sending the compiled page to a client that requested the page; and

accessing the set of static markup text in the shared, read-only memory when the code from the first instance of the application is executed;

wherein the method is performed by one or more computing devices.

6. (Currently Amended) A method according to claim 5, wherein the servlet class is not loaded into the shared, read-only memory when the other instances of the application are executed.

7. – 8. (Canceled)

9. (Currently Amended) A computer-implemented method according to claim 1, wherein:

the static markup text includes information to be displayed to a user and an annotation ~~directing~~ instructing a user agent how to render the information to be displayed to the user; and

the static markup text output by the executing servlet class includes the annotation.

10. (Canceled)

11. (Currently Amended) A method according to claim 5, wherein:

the set of static markup text includes information to be displayed to a user and an annotation ~~directing~~ instructing a user agent how to render the information to be displayed to the user; and

the set of static markup text output by the application includes the annotation.

12. – 14. (Canceled)

15. (Currently Amended) A non-transitory computer-readable storage medium ~~bearing~~ storing instructions that, when executed, cause one or more processors to perform a

method for sharing static markup text from a page among a plurality of users in response to requests from said plurality of users, said method comprising:

analyzing a page that includes static markup text and a set of code instructions executable on a server;

extracting the static markup text from the page and storing the static markup text in a resource file;

generating a servlet class for the page based on the set of code instructions, wherein the servlet class comprises a static initializer for initializing a static class variable for the servlet class, and wherein the servlet class does not include the static markup text;

in response to a first use of the servlet class by any instance of the servlet class:

invoking the static class initializer of the servlet class, wherein the invoking of the static class initializer of the servlet class causes the static markup text to be read from the resource file, and the static class variable to be initialized with the static markup text;

loading a copy of the static class variable initialized with the static markup text into shared memory;

in response to each request of a plurality of requests for the page from a plurality of clients, performing the steps of:

instantiating a distinct instance of the servlet class on the server, wherein instantiating each instance of the servlet class does not create another copy of the static markup text;

executing said distinct instance of the servlet class, wherein execution of each instance of the server class generates a compiled page ~~based on the copy of~~ by combining the static markup text that resides in the shared memory~~, and~~ with results produced by executing the set of code instructions; and

sending the compiled page to a client that requested the page.

16. (Canceled)

17. (Currently Amended) The non-transitory computer-readable storage medium of claim 15, wherein:

the static markup text includes information to be displayed to a user and an annotation ~~directing~~ instructing a user agent how to render the information to be displayed to the user; and

the static markup text output by the executing servlet class includes the annotation.

18. (Canceled)

19. (Currently Amended) A non-transitory computer-readable storage medium ~~bearing~~ storing instructions that, when executed, cause one or more processors to perform a method of initiating a first instance of an application that shares a set of static markup text with other instances of the application, wherein the first instance of the application is generated by compiling code from a page that contains both the code and the set of static markup text in response to a request from one or more users, said method comprising:

executing instructions to instantiate the first instance of the application;

wherein the instructions to instantiate the first instance of the application include:

analyzing the page to extract the set of static markup text and storing the set of static markup text in a resource file;

generating a servlet class for the page based on the code from the page, wherein the servlet class comprises a static initializer for initializing a set of static class variables for the servlet class, and wherein the servlet class does not include the static markup text;

in response to a first use of the servlet class by any instance of the servlet class:

invoking the static class initializer of the servlet class, wherein the invoking of the static class initializer of the servlet class causes the set of static markup text to be read from the resource file, and the set of static class variables to be initialized with the set of static markup text;

loading a copy of the set of static class variables initialized with the set of static markup text into shared, read-only memory;

in response to each request of a plurality of requests for the page from the one or more users, performing the steps of:

instantiating a distinct instance of the servlet class, wherein instantiating each

instance of the servlet class does not create another copy of the set of static class variables initialized with the set of static markup text;

executing said distinct instance of the servlet class, wherein execution of each instance of the server class generates a compiled page ~~based on the copy of~~ by combining the set of static markup text that resides in the shared, read-only memory~~, and~~ with results produced by executing the set of code instructions;

sending the compiled page to the user that requested the page; and

accessing the set of static class variables initialized with the set of static markup text in the shared, read-only memory when the code from the first instance of the application is executed.

20. (Currently Amended) The non-transitory computer-readable storage medium of claim 19, wherein the servlet class is not loaded into the shared, read-only memory when the other instances of the application are executed.

21. (Currently Amended) The non-transitory computer-readable storage medium of claim 19, wherein:

the set of static markup text includes information to be displayed to a user and an annotation ~~directing~~ instructing a user agent how to render the information to be displayed to the user; and

the set of static markup text output by the application includes the annotation.

22. (Canceled)

23. (Previously Presented) A computer-implemented method according to claim 1, wherein the servlet class includes an inner class.

24. (Currently Amended) A computer-implemented method according to claim 23, wherein the step of loading a copy of the static markup text includes hot-loading an instance of the inner class.

25. (Previously Presented) A computer-implemented method according to claim 24, wherein the inner class comprises an array of characters.

26. (Previously Presented) A method according to claim 5, wherein the servlet class includes an inner class.

27. (Currently Amended) A method according to claim 26 wherein the step of loading a copy of the set of the static markup text includes hot-loading an instance of the inner class.

28. (Previously Presented) A method according to claim 27, wherein the inner class comprises an array of characters.

29. (Currently Amended) A non-transitory computer-readable storage medium according to claim 15, wherein the servlet class includes an inner class.

30. (Currently Amended) A non-transitory computer-readable storage medium according to claim 29, wherein the step of loading a copy of the static markup text includes hot-loading an instance of the inner class.

31. (Currently Amended) A non-transitory computer-readable storage medium according to claim 30, wherein the inner class comprises an array of characters.

32. (Currently Amended) A non-transitory computer-readable storage medium according to claim 19, wherein the servlet class includes an inner class.

33. (Currently Amended) A non-transitory computer-readable storage medium according to claim 32, wherein the step of loading a copy of the set of the static markup text includes hot-loading an instance of the inner class.

34. (Currently Amended) A non-transitory computer-readable storage medium according to claim 33, wherein the inner class comprises an array of characters.